// Implement the C program for Page Replacement Algorithms: FCFS, LRU, andOptimal for frame size as

// minimum three

#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

#include <math.h>

#include <string.h>

// page replacement algorithms : FCFS

void fcfs(int n, int m, int a[n], int b[m])//a[]-->storing no of pages,b[]-->size of the frame

{

// i

int i, j, flag = 0, page\_fault = 0;

// use of page fault is to count the number of page faults

// use of flag is to check whether the page is present in the frame or not

for (i = 0; i < n; i++)

{

flag = 0;

for (j = 0; j < m; j++)

{

// if the page is present in the frame

if (a[i] == b[j])

{

// set the flag to 1

flag = 1;

break;

}

}

if (flag == 0)

{

// if the page is not present in the frame

// increment the page fault

b[i % m] = a[i];

page\_fault++;

}

}

printf("fcfs Page Faults : %d", page\_fault);

}

// page replacement algorithms : LRU

void lru(int n, int m, int a[n], int b[m])

{

int i, j, k, flag = 0, page\_fault = 0;

for (i = 0; i < n; i++)

{

flag = 0;

for (j = 0; j < m; j++)

{

// if the page is present in the frame

// set the flag to 1

if (a[i] == b[j])

{

flag = 1;

break;

}

}

// if the page is not present in the frame

// increment the page fault

if (flag == 0)

{

b[i % m] = a[i];

page\_fault++;

}

else

{

// if the page is present in the frame

// shift the page to the left

for (k = j; k < m - 1; k++)

{

b[k] = b[k + 1];

}

// set the last page to the current page

b[m - 1] = a[i];

}

}

printf("lru Page Faults : %d", page\_fault);

}

// page replacement algorithms : Optimal

void optimal(int n, int m, int a[n], int b[m])

{

int i, j, k, flag = 0, page\_fault = 0;

for (i = 0; i < n; i++)

{

flag = 0;

// if the page is present in the frame

// set the flag to 1

for (j = 0; j < m; j++)

{

if (a[i] == b[j])

{

flag = 1;

break;

}

}

// if the page is not present in the frame

// increment the page fault

if (flag == 0)

{

b[i % m] = a[i];

page\_fault++;

}

else

{

// if the page is present in the frame

// shift the page to the right

int max = -1;

for (k = 0; k < m; k++)

{

int f = 0;

for (j = i + 1; j < n; j++)

{

// if the page is present in the frame

// set the flag to 1'

if (b[k] == a[j])

{

if (j > max)

{

max = j;

f = 1;

}

break;

}

}

// if the page is not present in the frame

if (f == 0)

{

max = m;

break;

}

}

b[max] = a[i];

}

}

printf("Optimal Page Faults : %d", page\_fault);

}

// main function

int main()

{

printf("1. FCFS\n");

printf("2. LRU\n");

printf("3. Optimal\n");

printf("Enter choice : ");

int choice;

scanf("%d", &choice);

int n, m, i, j;

printf("Enter number of pages : ");

scanf("%d", &n);

printf("Enter number of frames : ");

scanf("%d", &m);

int a[n], b[m];

printf("Enter pages : ");

for (i = 0; i < n; i++)

{

scanf("%d", &a[i]);

}

for (i = 0; i < m; i++)

{

b[i] = -1;

}

switch (choice)

{

case 1:

fcfs(n, m, a, b);

break;

case 2:

lru(n, m, a, b);

break;

case 3:

optimal(n, m, a, b);

break;

default:

printf("Invalid choice");

}

return 0;

}